data entry boxes **570** and **576** do not match, no verification code is sent. The user then selects an acknowledgment softkey **582**, and a screen **584** (FIG. **15**F) is presented to the user. The user retrieves the verification code from his or her email account and then enters the verification code into a data entry box and submits the verification code to the remote server by activating softkey **588**. If the verification code entered into data entry box **586** matches the verification code sent by the remote server, then a screen **590** (FIG. **15**G) is presented to the user, which shows a list of backup files associated with the user **592**_a_, **592**_b_, **592**_c_, **592**_d_, **592**_e_. The user selects the backup file he or she wishes to restore to the electronic device and a screen **594** (FIG. **151**_f_) is presented, which presents to the user the security question selected at dropdown menu **564** selected when creating the backup file **592**_a_, **592**_h_, etc. selected. The user enters the answer to the question in a data entry box **596** and sends the answer to the remote server by selecting the softkey **598**. If the email entered in the data entry box **576**, the verification code entered in the data entry box **586**, and the answer entered into the data entry box **596** all match the corresponding prior data, then the remote server creates a decryption key, which is used to decrypt the backup file, and transmits the decrypted backup file to the electronic device over a secure communication channel. The backup file is then written to the electronic device, and then re-encrypted on the portable electronic device with the master password. The user may optionally choose to delete all records **24**, **26** already on the electronic device before the backup file is written to the electronic device by selecting an on/off box **599**.

Referring next to FIG. **16**, the programming to implement the asynchronous auto synchronization begins at a block **2052**, which checks to determine whether the application has been newly opened or, optionally, whether a particular time has elapsed from the last synchronization operation. If neither event has occurred, control pauses at the block **2052**; otherwise, control passes to a block **2054** that uploads the encrypted user data to the central server. The server checks the uploaded data to determine if such data are valid and, if so, the fields of the database maintained on the central server are updated for the particular records where data has changed since the last synchronization operation. A black **2058** then synchronizes remaining devices associated with the particular user.

Referring next to FIG. **17**, in the case of device theft or a lost device, or if the user operating the device is not authorized to have the information stored in his/her application, the owner of the application (and also the owner of the "subscription code") has the ability to issue a remote detonation of the mobile database or individual records stored in that database. When the application begins execution, a request is sent to the central server together with the active subscription code. If a remote detonation of data is requested by actuating a softkey **2070**, the server will respond with the command to "detonate" and optionally provide a list of records to delete. The application will automatically perform this detonation upon request.

More specifically, once the remote detonate softkey **2070** has been actuated, the software of FIG. **18** is executed by the system **10** (here shown as being initiated from the mobile device). The software begins at a block **2072**, which sends the detonation request and subscription code and device ID to the central server, (Note—see the question above regarding the device ID) Following the block **2072**, a block **2074** may receive an optional list of all records that may be marked for deletion on the lost or stolen device. A further block **2076** may optionally prompt the user to select records to delete and may further prompt the user to identify the device that has been lost or stolen, if such device has not been previously identified at the block **2072**. A block **2078** then commands the deletion of selected records on the identified device via the central server.

Referring next to FIG. **19**, in order to share password records between users, for example, between husband and wife or employees at a company, a mechanism for transmitting this password data in a secure and timely fashion is required. Additionally, in order for this feature to be used without burden, the feature needs to be simple and easy to use with minimal clicks. This may be accomplished, for example, by actuating a softkey **2090** in the Import/Export screen of the application that permits a first user to request the transmission of a record. The central server provides the user with a unique alphanumeric sequence of sufficient character length and a number of seconds for which the sequence will be valid (e.g., 60 seconds). Referring also to FIG. **20**, programming executed by the application running on the first user's device begins at a block **2092**, which awaits receipt of a unique alphanumeric sequence from the central server. Once this sequence is received, a block **2094** prompts the first user to identify a second user and record(s) to be sent to the second user. A block **2096** then encrypts the information to be sent with a cipher generated from the unique sequence and transmits the encrypted data to the central server.

The first user provides the unique alphanumeric sequence to the second user via telephone or text message. Referring to FIG. **21**, which illustrates programming executed by the application running on the second user's device, a block **2100** checks to determine whether a transfer request has been received from the central server. Once this occurs, a block **2102** prompts the second user to type in the unique alphanumeric sequence or accept the text message (SMS) into the second user's application, which enters the sequence on behalf of the second user. The sequence is then transmitted to the central server by a block **2104**. Thus, the application on the second user's device requests access from the central server to any information associated with the given sequence. If the information is found, and if the second user has requested the information within the time period allocated, the information is delivered to the second user in encrypted format. A block **2106** decrypts the record with the key as the cipher.

This method protects from brute force attack in addition to the cipher strength of the encryption/decryption algorithm chosen.

A specific embodiment of method and apparatus for protecting confidential information has been described for the purpose of illustrating the manner in which the invention is made and used. It should be understood that the implementation of other variations and modifications of the invention and its various aspects will be apparent to one skilled in the art, and that the invention is not limited by the specific embodiments described. Therefore, it is contemplated to cover the present invention and any and all modifications, variations, or equivalents that fall within the true spirit and scope of the basic underlying principles disclosed and claimed herein.

We claim:

1. An apparatus, comprising:

at least one storage device that stores a plurality of files, wherein each file contains at least one item of confidential information, and wherein a geographic location of use is associated with the file;

a position comparison processor coupled to the at least one storage device that compares a current geographic loca-